

CLAIMS

- 1 1. A runtime system for a global address space language for use with a
2 plurality of processors or computers, the system comprising:
3 a directory of shared variables comprising a data structure for tracking shared
4 variable information that is shared by a plurality of program threads; and
5 allocation and de-allocation routines for allocating and de-allocating shared
6 variable entries in the directory of shared variables.

- 1 2. The runtime system of claim 1 wherein the allocation and de-allocation
2 routines use pair-to-pair synchronization.

- 1 3. The runtime system of claim 1 wherein the runtime system is implemented
2 on a distributed memory system and the directory of shared variables is stored in a
3 private memory of each thread such that it is replicated across all of the threads.

- 1 4. The runtime system of claim 1 wherein the runtime system is implemented
2 on a shared memory system and the directory of shared variables is stored in a shared
3 memory shared by all threads.

- 1 5. The runtime system of claim 1 wherein the allocation and de-allocation
2 routines are used for both statically and dynamically allocated data.

- 1 6. The runtime system of claim 1 wherein arrays that are dynamically
2 allocated have affinity to a thread that called the allocation or de-allocation routine.

1 7. The runtime system of claim 1 wherein every thread has a handle for each
2 shared variable that it accesses.

1 8. The runtime system of claim 7 wherein the entries in the directory of shared
2 variables are accessed using the handle.

1 9. The runtime system of claim 7 wherein the handle comprises a partition
2 index and a variable index.

1 10. The runtime system of claim 1 wherein each thread has exclusive write
2 access rights to a partition of the directory of shared variables associated with the
3 thread.

1 11. A runtime system that scales to a plurality of processors for a global
2 address space language program having a plurality of threads that access memory in a
3 global address space system, the system comprising:

4 a shared data directory that maintains shared data entries related to shared data
5 structures that are shared by more than one of the plurality of threads; and

6 control structures to access, allocate and de-allocate the shared data structures
7 through the shared data directory.

1 12. The runtime system of claim 11 wherein the plurality of processors
2 operate as a shared memory machine.

1 13. The runtime system of claim 11 wherein the plurality of processor operate
2 as a distributed memory machine.

1 14. The runtime system of claim 11 wherein the shared data structures have
2 affinity to particular threads.

1 15. The runtime system of claim 11 wherein the shared data structures
2 comprise shared scalar variables, objects, arrays or pointers.

1 16. The runtime system of claim 15 wherein a shared scalar variable is
2 accessed by dereferencing a shared data directory partition for which the shared scalar
3 variable has affinity.

1 17. The runtime system of claim 15 wherein a shared array has a shared data

2 directory partition that points to a control structure that points to the shared array.

1 18. The system of claim 15 wherein the runtime system allocates a control
2 harness for a shared pointer when the shared pointer is declared by allocating a shared
3 control block and a shared address structure.

1 19. The system of claim 15 wherein some of the shared pointers have shared
2 targets and some of the shared pointers have private targets.

1 20. The system of claim 11 wherein entries to the shared data directory are
2 allocated by an owning thread or, in a synchronized manner by all threads at the same
3 time.

1 21. The runtime system of claim 11 comprising a handle that includes a
2 partition index and a variable index that is used by the threads to access the shared
3 variables.

1 22. The runtime system of claim 11 wherein the shared data directory includes
2 a partition that is used to access all statically declared non-scalar variables.

1 23. The runtime system of claim 11 wherein each thread uses a mutually
2 exclusive partition of the shared data directory.

1 24. A method of providing a scalable runtime system for a global address
2 space language, the method comprising:
3 creating a directory of shared variables containing information concerning data
4 shared by program threads for use by the threads in accessing the shared data; and
5 creating control structures to control allocation and de-allocation of the shared
6 data.

1 25. The method of claim 24 wherein creating control structures comprises
2 creating a plurality of control structures wherein each control structure controls the
3 allocation and de-allocation of a particular type of shared data structure.

1 26. The method of claim 24 comprising operating the runtime system on a
2 distributed memory machine.

1 27. The method of claim 26 wherein each thread contains a private copy of the
2 directory of shared variables and a calling thread allocates an entry in its directory of
3 shared variables and broadcasts an index of the entry to other threads.

1 28. The method of claim 26 wherein each thread has a private data control
2 structure with a pointer to a shared memory fraction.

1 29. The method of claim 24 comprising operating the runtime system on a
2 shared memory machine.

1 30. The method of claim 24 wherein a calling thread allocates space for a
2 shared variable and inserts a handle in a partition in the directory of shared variables.

3

4 31. The method of claim 29 wherein the control structures are common such
5 that any thread can access the common control structures.